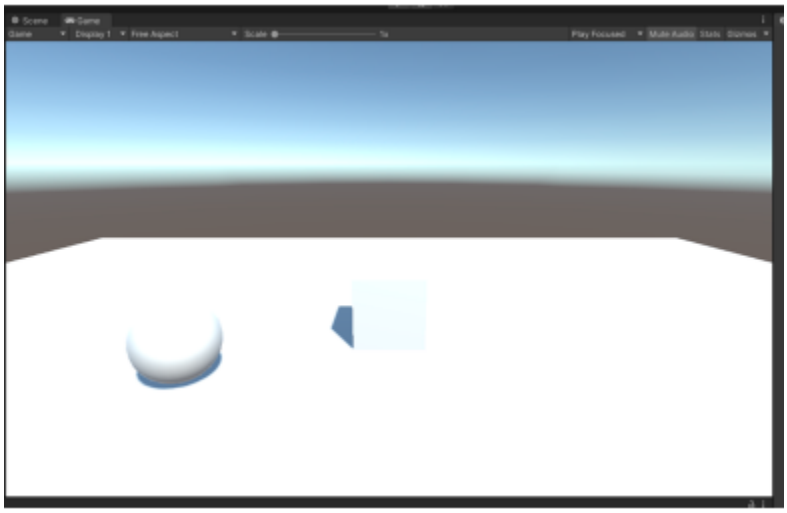
Experiment No. 4
Develop a scene in Unity that includes a cube, plane and sphere. Create a new material and texture separately for three Game objects. Change the color, material and texture of each Game object separately in the scene. Write a C# program in visual studio to change the color and material/texture of the game objects dynamically on button click
Date of Performance:
Date of Submission:
Marks:
Sign:

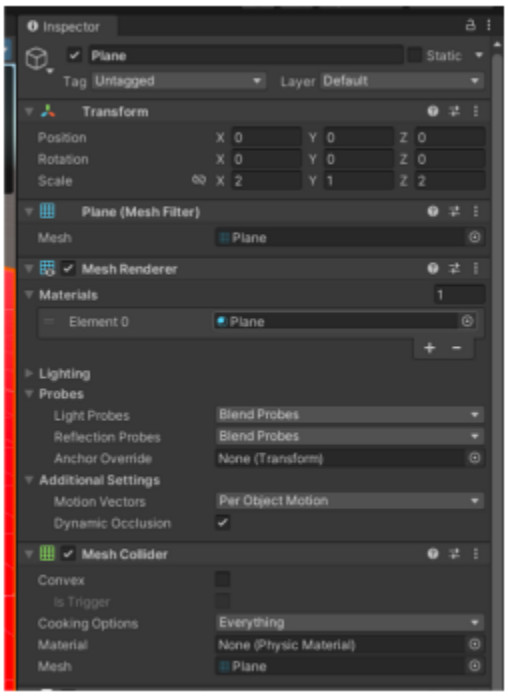
Aim: Develop a scene in Unity that includes a cube, plane and sphere. Create a new material and texture separately for three Game objects. Change the colour, material and texture of each Game object separately in the scene. Write a C# program in visual studio to change the colour and material/texture of the game objects dynamically on button click.

Theory:

Scene Creation:
1. Create a scene named Exp 4 in the similar manner to experiment 3 with the same folder structure
2. Create a cube, plane & sphere in the scene
3. Position them & the camera in such a way that all of them are visible in the Game view in the similar manner.



Material Creation:
1. Material is an asset that is used to define how a gameobject with a renderer component will look in the game.
2. Create a "Materials" folder inside the experiment 4 folder.
3. Create a material inside the folder by right click -> Create-> Material.
4. Click on the material & check if it has the standard shader assigned to it in the inspector.
5. Name the material "Plane". Create 2 other materials for cube & sphere.
6. Give a different color to each material from the inspector by changing the albedo.



Create Custom Components from code & Understanding Mono behaviour:

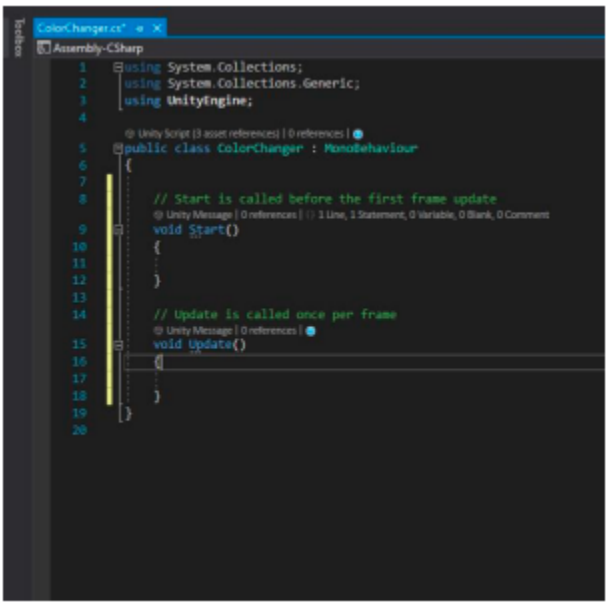
1. MonoBehaviour is the base class from which every Unity script derives.
2. If you need to add a script as a Component on a GameObject, you need to inherit the class from MonoBehaviour.
3. When a script is created the Start & Update functions are already present in it by default.

Editing the script as follows:

1.Remove the Update function from the script as we won't be needing that.
2. We need to implement the color changing functionality inside this script.
3. To do that we need a list of colors & textures that we will be changing on runtime at the button click.
4. Write the script as following:

1. Remove the Update function from the script as we won't be needing that.
2. We need to implement the color changing functionality inside this script.
3. To do that we need a list of colors & textures that we will be changing on runtime at the button click.
4. Write the script as following:

5. Save the script by Ctrl + S & switch to Unity.



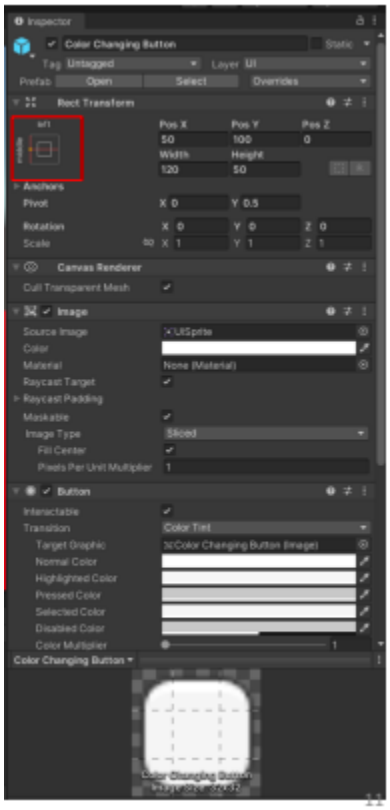
Creating User Interface (UI)

1. UI is the graphical interface where we can display information such as numbers, messages & add interactable buttons for the user to interact with.
2. Go back to Unity & in the hierarchy window, right click -> UI -> Canvas.
3. Now right click on the Canvas GameObject created & select UI -> Button - TextMeshPro.
4. Name the gameobject "Color Changing Button".

5. Click on the child text object of it & in the inspector in the text field type- "Switch Color".

Rect Transform Component

1. The Rect Transform component is the 2D layout counterpart of the Transform component. Where Transform represents a single point, Rect Transform represent a rectangle that a UI element can be placed inside.
2. All UI elements don't have a Transform but a RectTransform component.
3. It's used to store and manipulate the position, size, and anchoring of a rectangle and supports various forms of scaling based on a parent RectTransform.
4. We have to position the button from the left side of the screen so we will do that by clicking on the box shaped icon on the top left corner of the RectTransform component in the inspector. Then press shift+alt on the keyboard & click on the 1st icon from the middle row in the grid.

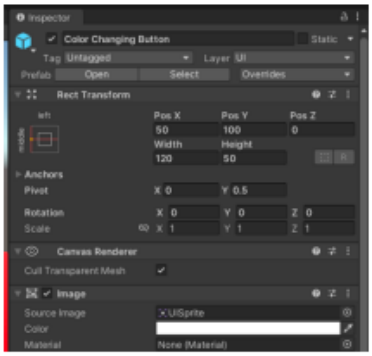


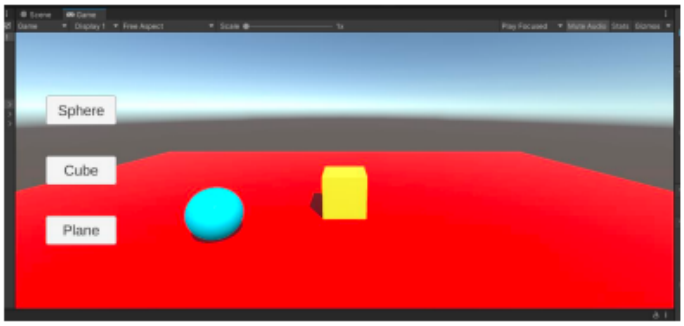
Prefabs:
1. Unity's Prefab system allows you to create, configure, and store am GameObject complete with all its components, property values, and child GameObjects as a reusable Asset.
2. The Prefab Asset acts as a template from which you can create new Prefab instances in the Scene.
3. When you want to reuse a GameObject configured in a particular way –like a non-player character (NPC), prop or piece of scenery – in multiple places in your Scene, or across multiple Scenes in your Project, you should convert it to a Prefab.

4. This is better than simply copying and pasting the GameObject, because the Prefab system allows you to automatically keep all the copies in sync.

Creating Button Prefab:
1. To a create a Prefab, simply drag & drop a gameobject from the hierarchy to a folder in the project window.
2. Drag & drop the Color changing button from the hierarchy into a "Prefabs" folder inside the Experiment 4 folder.

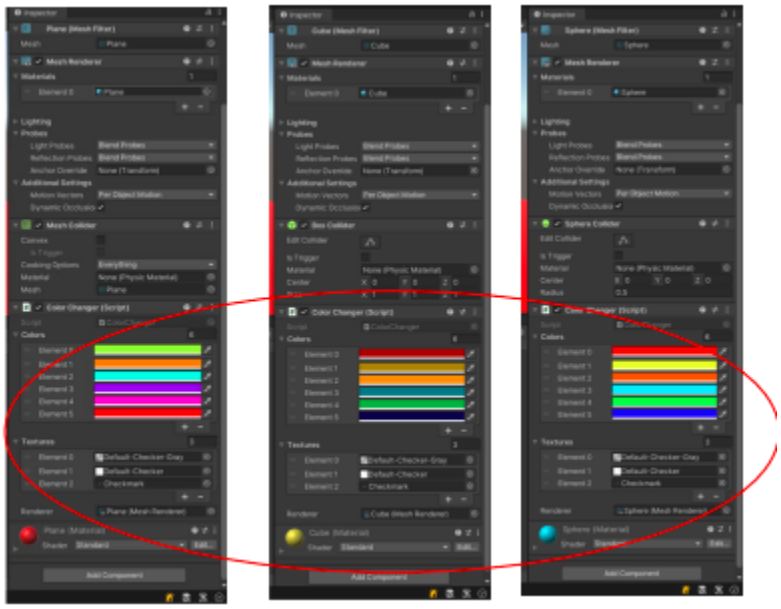
Arranging the buttons:
1. We will be needing the buttons for changing the colors of 3 GameObjects so create 2 more duplicates of the button GameObject in hierarchy view by clicking on it & pressing Ctrl + D.
2. We will need to reposition the buttons so they don't overlap.
3. Edit the "Pos Y" of the RectTransform of 2 buttons to make them position in a similar way.
4. Click on the arrow besides the Button gameObject in the hierarchy to expand it. Then Change the text of the buttons by selecting the Text gameobject of the button child & Changing the text in the inspector.





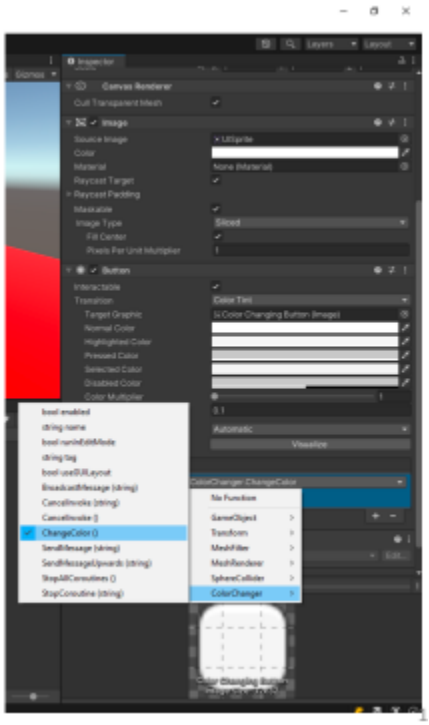
Setting up Color Changes:

1. Select the Plane, Cube, Sphere in the Hierarchy & in the inspector click on Add component.
2. Search for Color Changer Script & press Enter.
3. Setup each color changer of each gameobject individually.
4. For instance: Setup the Plane by clicking on it. In the Inspector window to add colors in the Colors List field, click on the plus icon on the right of the field. Click on the new color added & select the color from the palette & close the palette window.
5. Similarly add Textures to the Textures list.

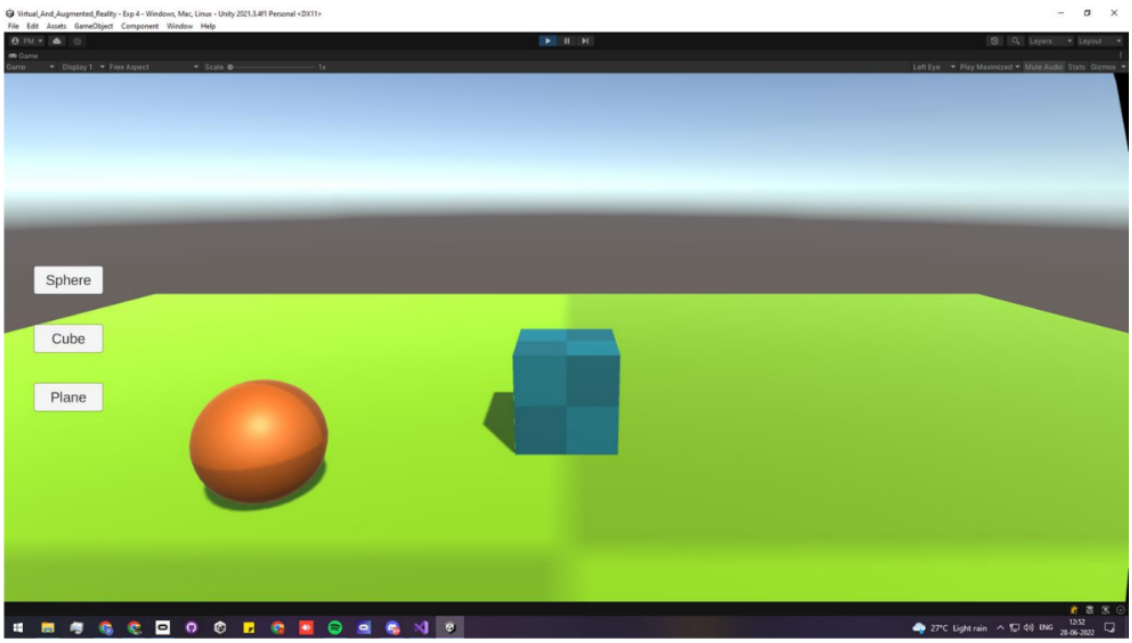


Calling Functions:

1. To call a script's function from a button click click on a button.
2. In the Inspector, look at the OnClick Event of the Button component.
3. Click on the + sign on the bottom right of the Event.
4. Drag & drop one of the 3 GameObjects on the field written as "None (Object)". Then from the dropdown menu select the ColorChanger script & then the ChangeColor() function.
5. Setup the other 2 buttons too for the other Color Changing GameObjects in a similar way.



Output:



Conclusion:-

In Unity, we successfully created a dynamic scene featuring a cube, plane, and sphere, with unique materials and textures assigned to each game object. The experiment allowed us to change the color, material, and texture of each object independently within the Unity scene, offering a valuable demonstration of real-time interactivity. We further implemented a C# program in Visual Studio to enable dynamic changes in color, material, and texture for these game objects upon button click. This project highlights the versatility of Unity in creating visually appealing and interactive environments and serves as a foundational example for more complex real-time content manipulation in Unity applications.